A Study on Geometry Contrast Enhancement for 3D Point Models

Jin-Woo Nam

Electrical Engineering Program

Graduate school of UNIST

2012

A Study on Geometry Contrast Enhancement for 3D Point Models

Jin-Woo Nam

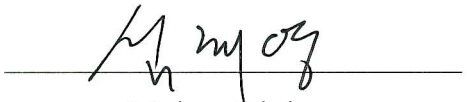
Electrical Engineering Program
Graduate School of UNIST

A Study on Geometry Contrast Enhancement for 3D Point Models

A thesis
submitted to the Graduate School of UNIST
in partial fulfillment of the
requirements for the degree of
Master of Science

Jin-Woo Nam

6. 8. 2012 Month/Day/Year of submission
Approved by



Major Advisor
Jae-Young Sim

A Study on Geometry Contrast Enhancement for 3D Point Models

Jin-Woo Nam

This certifies that the thesis of Jin-Woo Nam is approved.

6. 8. 2012 Month/Day/Year of submission

Thesis Supervisor: Jae-Young Sim

Seung-Joon Yang: Thesis Committee Member #1

Won-Ki Jeong: Thesis Committee Member #2

Abstract

Point primitives have come into the spotlight as a representation method of 3D models. A lot of researches have been performed on the modeling, processing, and rendering 3D point models. Especially, various methods have been developed for the extraction and preservation of the salient features of corners, curves, and edges in 3D point models. However, little effort has been made to extract and enhance the weak features that are relatively imperceptible due to the low geometry contrast. In this thesis, we propose a novel method to improve the visibility of 3D point models by enhancing the geometry contrast of weak features. We first define a weak feature region as a group of local points yielding small deviations of normal directions. Then we define the geometry histogram for each region as the distribution of the signed distance between a feature point and the locally approximated plane. We equalize and stretch the geometry histogram and move the corresponding feature points accordingly. We also render the enhanced model using the normal mapping for better visual presentation. Experimental results demonstrate that the proposed method enhances the geometry contrast of 3D point models by refining the appearance of the weak features. We expect that the geometry contrast enhancement algorithm will facilitate many applications in various fields.

Contents

List of Figures

List of Tables

Nomenclature

`API` Application Program Interface. 4

`CAD` Computer-Aided Design. 8

`CPU` Central Processing Units. 10

`CT` CAT Scan. 1

`CUDA` Compute Unified Device Architecture. 11

`FPS` Frame Per Second. 29

`GPU` Graphics Processing Units. 4, 10

`HDTV` High Definition Television. 1

`KNN` K-Nearest Neighbors. 28

`MST` Minimum Spanning Tree. 14

`PCA` Principal Component Analysis. 8, 17

`PCL` Point Cloud Library. 29

`PDA` Personal Digital Assistant. 1

`PDF` Probability Density Function. 21

Chapter I

Introduction

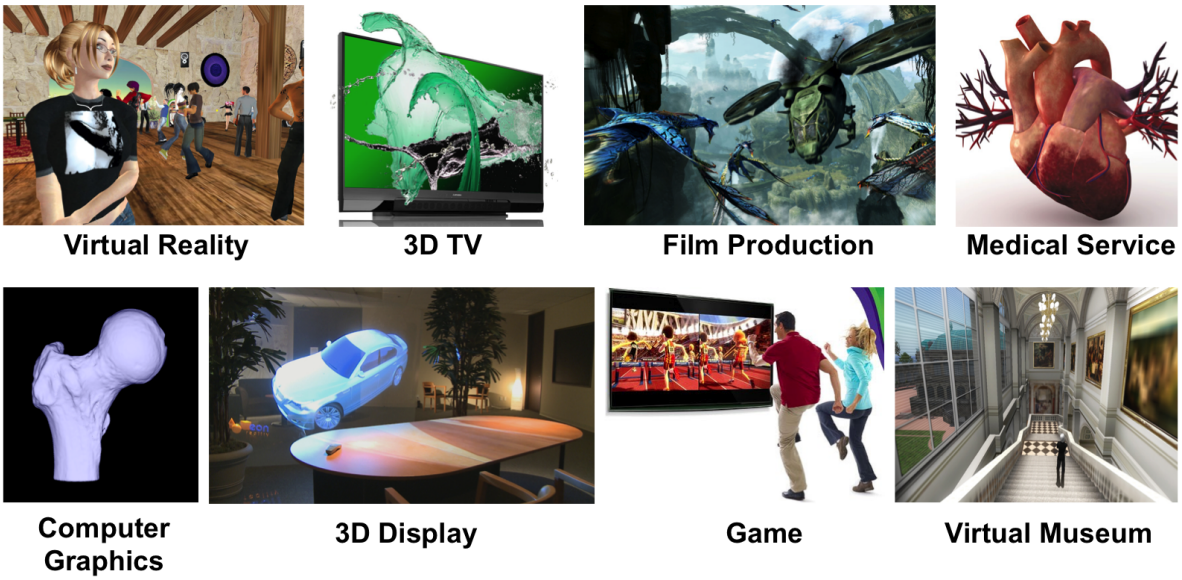
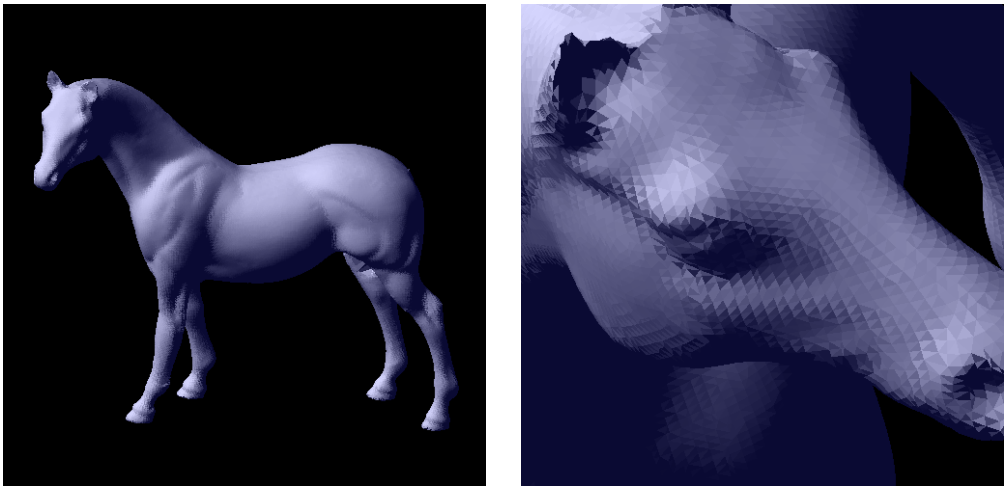


Figure 1.1: Applications based on 3D visual media.

In recent years, the desire to utilize visual media without the constraint of time or location has been increasing. Especially, the applications of 3D visual media can facilitate more realistic and immersive visual experiences. As shown in Figure 1.1, many applications using 3D visual media technologies are being actively developed in various fields: 3D visual communications, 3D broadcasting using , film production, biomedical imaging, computer graphics, display, and games. Moreover, there are also some projects that apply 3D visual media to personal electronic devices such as mobile phones.

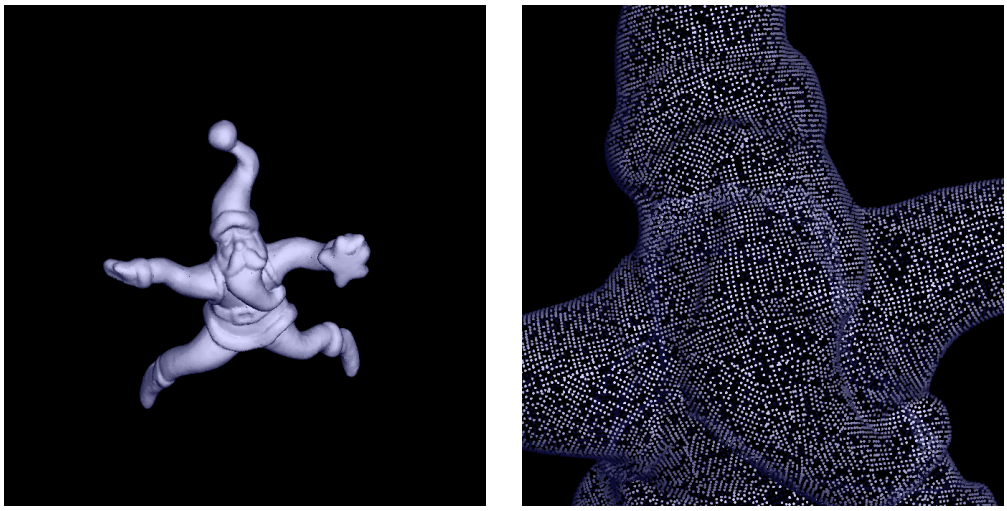
With the increasing popularity of 3D visual media, several researches have been performed to efficiently represent and model the 3D visual data. For example, polygonal mesh representation, which is a collection of vertices, edges, and faces, has been considered as one of the most common representations of 3D visual data in computer graphics. Figure 1.2 shows an example of the rendered 3D polygonal mesh model. However, the polygonal mesh representation spends a large number of polygons to represent a highly detailed 3D model, which requires a huge computational complexity in rendering.



(a) (b)

Figure 1.2: 3D polygonal mesh model. (a) The "Horse" model. (b) A close-up view.

Point primitives (also known as point clouds) have been introduced as an alternative tool for representation of 3D visual data (Levoy 1999). The 3D point model yields no connectivity information, and thus each point is directly rendered. Furthermore, a set of points can be acquired using inexpensive scanning devices nowadays. Figure 1.3 shows a rendered 3D point model.



(a) (b)

Figure 1.3: 3D point model. (a) The "Santa" model. (b) A close-up view.

Due to the simplicity of point representations, a lot of researches have been performed to investigate the applications of 3D point models in various fields. One of the researches is feature analysis for 3D point models (Fang et al. 2006, Weber et al. 2010). Most works focus on the extraction of salient features, such as edges and corners, from a 3D point model. However, little effort has been made to handle inconspicuous features, which usually degrade the visual perception of a rendered 3D point model.

(a) (b)

Figure 1.4: Image contrast enhancement. (a) A low contrast image. (b) An enhanced image.

In this thesis, we improve the visibility of 3D point models by extracting and enhancing the weak features. We employ the histogram equalization technique, which is a widely used technique for contrast enhancement of images (Gonzalez & Woods 2007). Figure 1.4 shows a concept of histogram equalization. A low contrast image has a narrow range of intensity values, as shown in Figure 1.4a. By uniformly distributing the image histogram which represents the number of pixels of each intensity value, we can get an image with a more enhanced contrast, as shown in Figure 1.4b.

Based on the concept of the histogram equalization technique, our method can be described as follows. For a 3D point model, we first detect the weak feature points by using the normal deviation between neighboring points. The weak feature points yield small but distinct normal deviation. Then the selected points are grouped into several local regions. For each weak feature region, we define a geometry histogram as the distribution of the signed distances of points from the basis plane. By equalizing and stretching the geometry histogram, we obtain a model with a higher geometrical contrast.

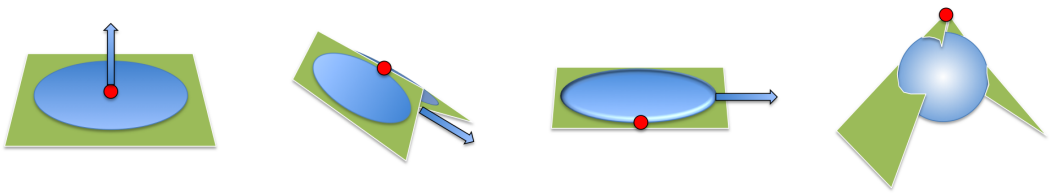
The remainder of this thesis is organized as follows. Chapter 2 presents the related works. Chapter 3 describes the proposed method for improving the geometrical contrast of 3D point model based on the histogram modification. Chapter 4 presents the experimental results. Finally, concluding remarks are given in Chapter 5.

Chapter II

Related Works

We review the feature analysis techniques for 3D point models.

Gumhold et al. present a method to extract linear features from a 3D point model (Gumhold et al. 2001). They first conduct a principal component analysis using the neighboring points of each point. The resulting eigenvalues $(\lambda_1, \lambda_2, \lambda_3)$ are then used to measure the likelihood that the point deserves to become a feature. As shown in Figure 2.1, the ellipsoid is formed by the eigenvalues and eigenvectors, which distinguishes the corner and border points from feature (crease) points. Since the initial feature points are quite dense, Gumhold et al. produce clear lines by computing a minimal spanning tree on a subset of the neighboring points.

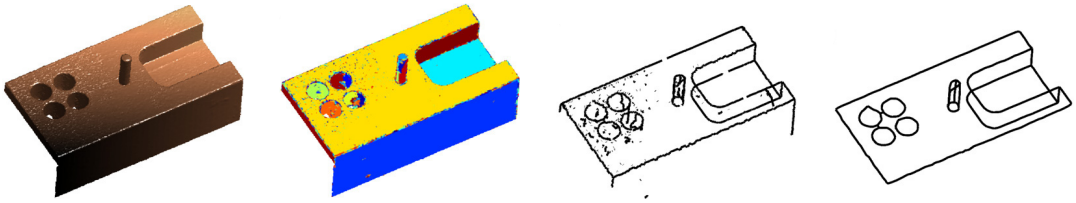


(a) (b) (c) (d)

Figure 2.1: Point classification using the three eigenvalues $(\lambda_1 \leq \lambda_2 \leq \lambda_3)$ obtained by principal component analysis (Gumhold et al. 2001). (a) Surface point $[\lambda_1 \approx 0, \lambda_2 \approx \lambda_3]$. (b) Crease point $[\lambda_1 \approx \lambda_2, \lambda_1 + \lambda_2 \approx \lambda_3]$. (c) Border point $[\lambda_1 \approx 0, 2\lambda_2 \approx \lambda_3]$. (d) Corner point $[\lambda_1 \approx \lambda_2 \approx \lambda_3]$.

Pauly et al. introduce a technique to detect feature lines with a multi-scale analysis of local neighborhoods (Pauly et al. 2003). They also use the principal component analysis for the classification of each point in a 3D point model. They define a surface variation that corresponds to the ratio of the smallest eigenvalue to the sum of all eigenvalues as a weight, that measures the extent to which the point belongs to a feature. Moreover, Pauly et al. apply their feature detection procedure to a multi-scale approach that varies the size of the neighborhood.

Demarsin et al. focus on producing closed sharp feature lines in a 3D point model (Demarsin et al. 2007). They use a region-growing method that segments a point cloud into clusters based on normal similarity, in order to identify feature regions that have large normal variation (refer to Figure 2.2b). A graph



(a) (b) (c) (d)

Figure 2.2: Extraction procedure for closed feature lines (Demarsin et al. 2007). (a) Original point model. (b) Segmented clusters. (c) Pruned minimal spanning graph. (d) Final feature lines.

approach is then used to connect the neighboring clusters. Since an edge between clusters that have large normal variation is possibly a feature, the edges in this graph are used as an indicator for the existence of sharp features. Demarsin et al. construct an initial feature by assigning a small weight and computing a minimal spanning tree on a graph that is similar to Pauly et al. (Pauly et al. 2003). The distance and the path lengths between two unlinked points in the graph are then used to determine whether the two points should be linked or not. Afterward, the branches of any remaining unlinked points are cut (refer to Figure 2.2c), and the graph is smoothed to obtain the final closed feature lines (refer to Figure 2.2d).

Chapter III

Geometry Contrast Enhancement for 3D Point Models

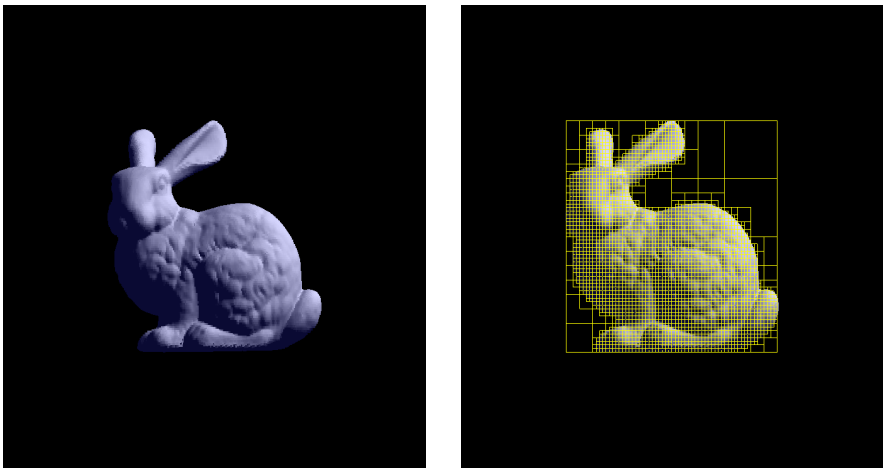
3.1 Overview of the Proposed Algorithm

Figure 3.1: Flowchart of the proposed algorithm.

Figure 3.1 summarizes the logical flow of the proposed algorithm. An unorganized 3D point model is first preprocessed to construct a connectivity and estimate normal directions. We can optionally use the GPU (Graphics Processing Unit) in this stage for efficient computing. After preprocessing, we define and extract the weak feature points from a 3D point model using the normal deviations of locally neighboring points. Then, we group the nearby points together into several homogeneous weak feature regions. For each weak feature region, we introduce a novel concept of geometry histogram which is then equalized and stretched to move the weak feature points for enhancement of geometry contrast. The modified point model can be directly rendered. In addition, we also propose a normal mapping (Blinn 1978, Cohen et al. 1998) based rendering, which preserves the original point positions but changes the normal directions for better visual presentation.

3.2 Preprocessing

As mentioned before, the connectivity information and normal information are not available in 3D point models. In order to extract the weak feature points based on the normal directions, we first estimate a normal direction at each point. We take an input 3D point model, in which the points are densely sampled by a 3D scanning equipment. In order to efficiently handle such large number of points, 3D data structures such as KD-tree (Bentley 1975) or octree (Jackins & Tanimoto 1980) are usually employed.



(a) (b)

Figure 3.2: Octree data structure. (a) The "Stanford bunny" Model. (b) Octree partitioning.

We employ the octree structure as shown in Figure 3.2.

Then we compute a normal vector at each point $\mathbf{p}_i \in \boldsymbol{P}$ by analyzing the local neighborhood characteristics around $\mathbf{p}_i$. As seen in Figure 3.3, we estimate the normal vector $\overrightarrow{\mathbf{n}_i}$ at $\mathbf{p}_i$ by performing PCA (principal component analysis) using the covariance matrix $\mathbf{C}$ of k neighboring points of $\mathbf{p}_i$, which is given by

$$\mathbf{C} = \begin{bmatrix} \mathbf{p}_{i,1} - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i,k} - \overline{\mathbf{p}} \end{bmatrix}^{\mathbf{T}} \begin{bmatrix} \mathbf{p}_{i,1} - \overline{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i,k} - \overline{\mathbf{p}} \end{bmatrix} \tag{3.1}$$

where $\overline{\mathbf{p}}$ denotes the centroid of the k neighboring points $\mathbf{p}_{i,j}, j \in (1, \cdots, k)$ of $\mathbf{p}_i$. We obtain the pair $(\mathbf{V}_l, \lambda_l)$, where $\mathbf{V}_l$ is the eigenvector and λ_l is the eigenvalue, based on the equation 3.2

$$\mathbf{C} \cdot \mathbf{V}_l = \lambda_l \cdot \mathbf{V}_l, \quad l \in [0, 1, 2]\,. \tag{3.2}$$

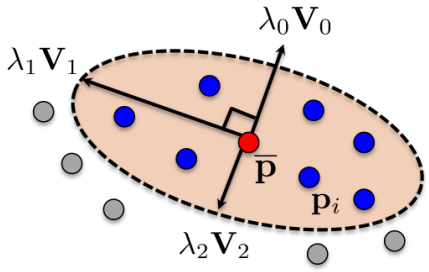


Figure 3.3: Illustration of covariance analysis. A normal vector $\overrightarrow{\mathbf{n}_i}$ at the point $\mathbf{p}_i$ is defined as the eigenvector $\mathbf{V}_l$ corresponding to the smallest eigenvalue λ_l.

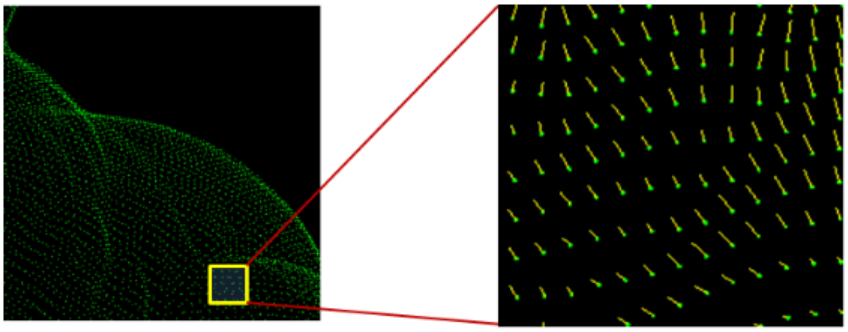


Figure 3.4: Consistent normal orientation. The yellow lines denote the directions of normal vectors.

According to the fundamental eigenanalysis theorem, the eigen-pair $(\mathbf{V}_l, \lambda_l)$ can be found by the following procedure. First, we get the eigenvalues λ_l's from the 3rd order polynomial equation, $\det(\mathbf{C} - \lambda\mathbf{I}) = 0$. Then, we find the eigenvector for a given λ_l by employing Gauss-Jordan elimination method to the set of homogeneous linear equations $(\mathbf{C} - \lambda_l\mathbf{I})\,\mathbf{V}_l = 0$. Since the eigenvalue λ_l denotes the scale of the corresponding eigevector $\mathbf{V}_l$, we can consider λ_l as the variation of the neighboring points $\mathbf{p}_{i,j}$ according to the direction of $\mathbf{V}_l$. Therefore, when we order the resulting eigenvalues as $\lambda_0 \leq \lambda_1 \leq \lambda_2$, the eigenvector $\mathbf{V}_0$ corresponding to the smallest eigenvalue can be regarded as the normal vector $\overrightarrow{\mathbf{n}_i}$ at $\mathbf{p}_i$.

The estimated normal directions should be consistently oriented. We first select a seed normal $\overrightarrow{\mathbf{n}_a}$ at the point $\mathbf{p}_a$ which is correctly directed. Then we propagate the normal orientation of $\overrightarrow{\mathbf{n}_a}$ from $\mathbf{p}_a$ to its adjacent point $\mathbf{p}_b$ along the most plausible path (Hoppe et al. 1994). Specifically, a cost, which measures the difference between $\overrightarrow{\mathbf{n}_a}$ and $\overrightarrow{\mathbf{n}_b}$, is assigned to each edge $\mathbf{e}_{a,b}$ by traversing the spanning tree of a point model. Figure 3.4 shows an example of consistently oriented normal directions.

3.3 Weak Feature Extraction

3.3.1 Selection of Weak Feature Points

The most conventional algorithms of feature analysis for 3D point models consider the detection of the salient features such as edges or corners. However, we focus on the indistinct weak features. While the salient features are obviously perceived, the weak features are relatively hard to be recognized which degrades the visibility of 3D point models.

Let us first consider feature analysis in image processing. Table 3.1 shows several feature extraction methods in image processing. A pixel is determined as a feature when it belongs to edges or curves based on the difference of intensity values from the neighboring pixels (Nixon & Aguado 2008). For example, the vertical edge is obtained from a large difference of pixel values along the horizontal direction. In contrary, the horizontal edge is determined by comparing pixel values along the vertical direction.

We employ the normal direction at each point to determine the feature points in a 3D point model. Under the assumption that we already precompute the connectivity information for a 3D point model, we take the first-order differencing operation for the normal vectors of the neighboring points. Intuitively, we

Topic	Detection Methods
First-order detection	Roberts, Prewitt, Sobel, Canny
Second-order detection	Laplacian, Marr-Hildreth, LoG
Other edge operators	Spacek, Petrou.
Detecting image curvature	Curve fitting, Intensity change, Harris corner detector
Optical flow estimation	Horn and Schunk method

Table 3.1: Low-level feature extraction in image processing (Nixon & Aguado 2008).

classify the point $\mathbf{p}_i$ into the three surface patterns according to the normal difference. First, the normal directions of neighboring points on a smooth surface are almost similar to each other as illustrated in Figure 3.5a. Second, the normal directions of adjacent points on a rough surface are obviously different to each other as shown in Figure 3.5b. These two surface patterns are mostly studied in the conventional feature analysis algorithms. However, we introduce the third surface pattern of weak feature which represents the shallowly engraved or slightly protruding patterns as illustrated in Figure 3.5c. In such a case, the deviation of normal directions of neighboring points are relatively small.

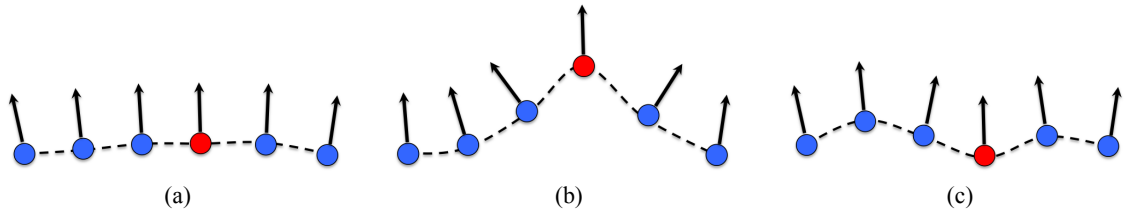


Figure 3.5: Illustration of surface patterns. (a) Smooth surface. (b) Strong feature of rough surface. (c) Weak feature.

The normal deviation $\sigma(\mathbf{p}_i)$ at $\mathbf{p}_i$ is computed as the average angle between the two normal vectors of $\overrightarrow{\mathbf{n}_i}$ and $\overrightarrow{\mathbf{n}_{i,j}}$, as

$$\sigma(\mathbf{p}_i) = \frac{1}{K}\sum_{j=1}^{K} \arccos\left(\frac{\overrightarrow{\mathbf{n}_i} \cdot \overrightarrow{\mathbf{n}_{i,j}}}{||\overrightarrow{\mathbf{n}_i}||\ ||\overrightarrow{\mathbf{n}_{i,j}}||}\right) \tag{3.3}$$

where K denotes the number of neighboring points of the $\mathbf{p}_i$. We employ the 8-nearest neighbor

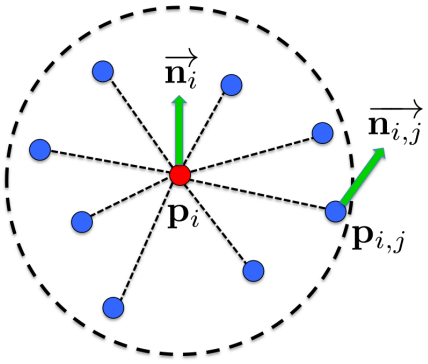
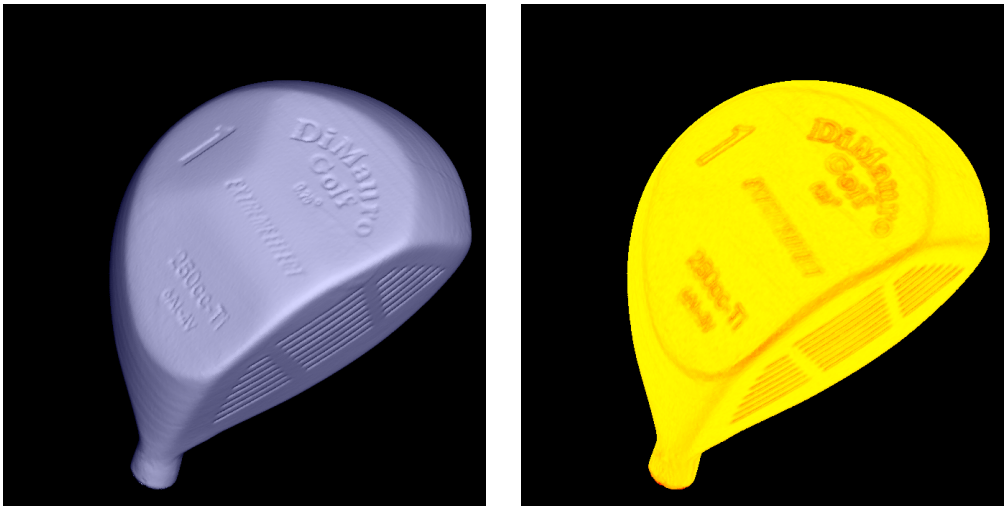


Figure 3.6: Normal deviation from the neighboring points.



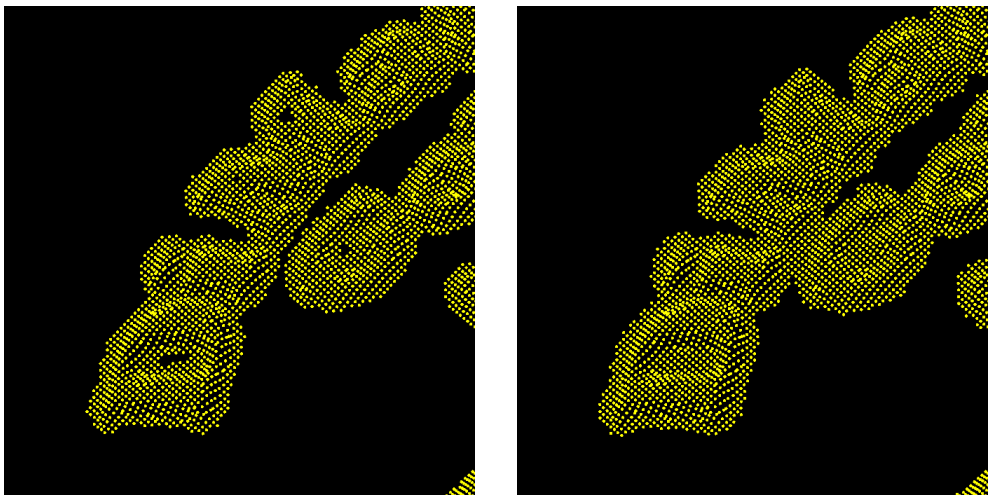
(a) (b)

Figure 3.7: Normal deviation on the "Golf-club" model. (a) Original "Golf-club" model. (b) Color-coded deviation. The yellow corresponds to low values and the red to high values.

points to compute $\sigma(\mathbf{p}_i)$ as shown in Figure 3.6. We select $\mathbf{p}_i$ as a weak feature point, when $\sigma(\mathbf{p}_i)$ belongs to a specific user-defined range, which is adaptively set according to a 3D point model. Figure 3.7 illustrates an example of the strength of normal deviation of the "Golf-club" model. The red and yellow colors represent the large and small deviation values, respectively.

3.3.2 Region Growing based Feature Extraction

As shown in Figure 3.8a, we can observe some holes and crevices on weak feature regions. These missing points may affect on the further geometry processing. To solve this problem, we make up the holes by using the region growing algorithm in a flood-fill algorithm manner.



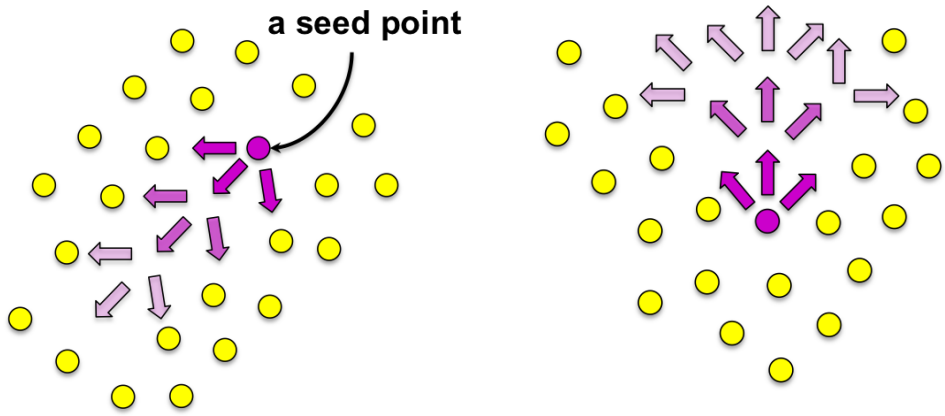
(a) (b)

Figure 3.8: Weak feature extraction with region growing. (a) Weak feature regions that have several crevices and holes. (b) The resulting weak feature regions.

In order to generate connected homogeneous regions of weak features, we first apply the mean filtering for the values of $\sigma(\mathbf{p}_i)$ as

$$\sigma(\mathbf{p}_i) = \frac{1}{K} \sum_{m=0}^{K} \sigma(\mathbf{p}_m). \tag{3.4}$$

Then a weak feature point is selected as a seed point, and we check its neighboring points whether they are weak feature points or not, as shown in Figure. 3.9. If the neighboring points are also weak feature points, then they are included into a feature region and serve as the next seed points for region growing. This process is iteratively performed until no more feature points are found. The result of region growing is shown in Figure 3.8b.



(a) (b)

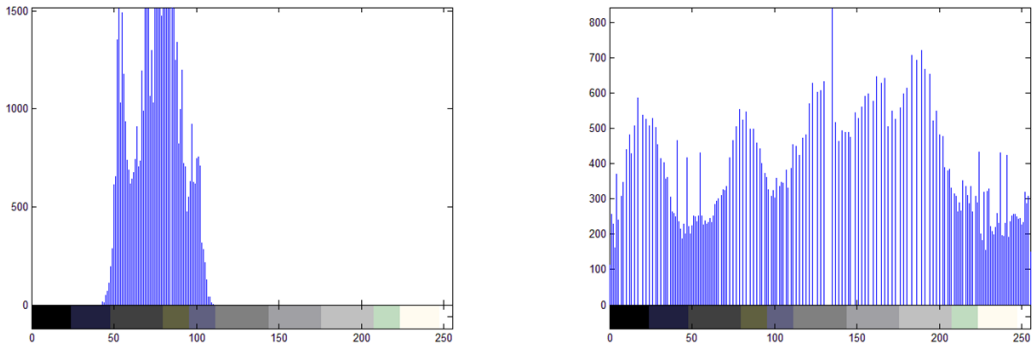
Figure 3.9: Illustration of the region growing process applied to our weak feature extraction. (a) In case of holes. (b) In case of crevices. The yellow denotes a point that belongs to a weak feature while the purple means a seed point.

3.4 Geometry Histogram Modification

In this thesis, we define a concept of geometry contrast as the differences in geometric characteristics of features. Since the weak features of a 3D point model generally yield a low geometry contrast, they are hardly distinguishable from the background surface. Our geometry contrast enhancement framework is originated from the motivation of contrast enhancement techniques for images. We first review the generic image contrast enhancement process. Then, we examine how the concept of image histogram modification can be applied to the geometry histogram modification, commenting on the fundamental differences between images and 3D point models.

3.4.1 Histogram modification of Images

Histogram modification techniques are used to appropriately adjust the contrast of an image by changing the histogram distribution of an image (Gonzalez & Woods 2007). For example, the histogram distribution of Figure 1.4a has a narrow range of intensity levels. By applying histogram equalization technique to the image, we make the image histogram more uniformly distributed and lead to the enhanced image as shown in Figure 3.10b.



(a) (b)

Figure 3.10: Histogram equalization. (a) Original histogram of Figure 1.4a. (b) Equalized histogram (refer to Figure 1.4b). The x-axis and y-axis denote the intensity level $k \in [0, 255]$ and the number of pixels, respectively.

Let us describe the histogram equalization in more detail. We suppose that the discrete intensity values in an image have the PDF (Probability Density Function) $\mathbf{P_r}(\boldsymbol{I}_k)$,

$$\mathbf{P_r}(\boldsymbol{I}_k) = \frac{n_k}{M},\ k \in (0, 1, \cdots, R-1) \tag{3.5}$$

where M denotes the total number of pixels in an image, n_k indicates the number of pixels that have the intensity value $\boldsymbol{I}_k$, and R is the number of possible intensity levels. Then the transformation function $\mathbf{T}$ is given as

$$\mathbf{T}(\mathbf{I}_k) = \frac{R-1}{M} \sum_{j=0}^{k} n_j. \tag{3.6}$$

The intensity level $\boldsymbol{I}_k$ in the input image is mapped into the intensity level $\mathbf{T}(\boldsymbol{I}_k)$ in the resulting image, using Equation 3.6.

3.4.2 Geometry Histogram

We apply the concept of histogram modification of images to 3D point models. We first define the geometry histogram for each weak feature region, as illustrated by Figure 3.11. We evaluate the signed

distance d from the average position $\overline{\mathbf{p}}$ of a weak feature region to a point $\mathbf{p}$ along the specified direction $\overrightarrow{\mathbf{e}}$,

$$d = (\mathbf{p} - \overline{\mathbf{p}}) \cdot \overrightarrow{\mathbf{e}} \tag{3.7}$$

where $\overrightarrow{\mathbf{e}}$ is defined as the histogram direction vector and basically changeable. We define the directional geometry histogram as the probability distribution of d associated with the points in a weak feature region. Therefore, for the same group of points, the geometry histogram yields the different distributions according to the selection of histogram direction vector. We set $\overrightarrow{\mathbf{e}}$ as the normal vector of the plane which approximates the distribution of $\mathbf{p}$'s in a weak feature region and passes through $\overline{\mathbf{p}}$, since $\mathbf{p}$'s yield the smallest variation along this direction.

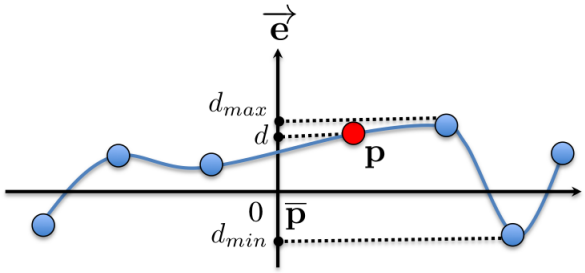


Figure 3.11: Distribution of points in a weak feature region.

3.4.3 Geometry Histogram Modification

Figure 3.12 illustrates the geometry contrast enhancement which moves the position of an original point $\mathbf{p}$ to a new position $\widetilde{\mathbf{p}}$ along the corresponding histogram direction $\overrightarrow{\mathbf{e}}$ via

$$\widetilde{\mathbf{p}} = \mathbf{p} + (\widetilde{d} - d) \cdot \overrightarrow{\mathbf{e}}, \tag{3.8}$$

where $\widetilde{d} = (\widetilde{\mathbf{p}} - \overline{\mathbf{p}}) \cdot \overrightarrow{\mathbf{e}}$. Specifically, the new distance $\widetilde{d}$ is obtained from d by the following transform of histogram modification

$$\widetilde{d} = (\widetilde{d}_{max} - \widetilde{d}_{min})\boldsymbol{Y}(\boldsymbol{d}) + \widetilde{d}_{min}, \tag{3.9}$$

which makes the geometry histogram of $\widetilde{d}$ yield a more uniform distribution than that of d. $\boldsymbol{Y}(\boldsymbol{d})$ represents the cumulative distribution function of d, which is used to equalize the geometry histogram. Practically, we compute

$$\boldsymbol{Y}(\boldsymbol{d}) = \frac{\text{number of points with distances less than } d}{\text{total number of points}}, \tag{3.10}$$

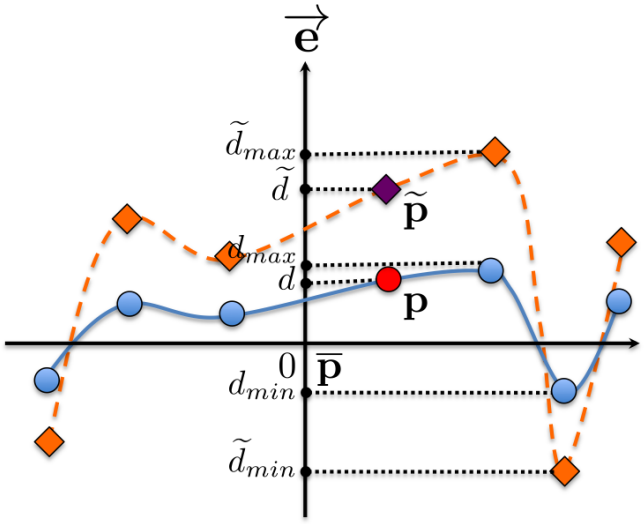


Figure 3.12: Geometry contrast enhancement using histogram equalization and stretching along the histogram direction. The solid and dashed curves denote the original and modified geometry, respectively.

in a selected weak feature. As shown in Figure 3.12, the original positions of neighboring points are not clearly distinguishable from each other along the direction $\vec{\mathbf{e}}$, since the geometry histogram of d yields a compact distribution within a relatively narrow range of $[\widetilde{d}_{min}, \widetilde{d}_{max}]$, and thus yield a higher geometry contrast along the histogram direction.

$\widetilde{d}_{min}$ and $\widetilde{d}_{max}$ in Equation 3.9 denote the lower bound and the upper bound of $\widetilde{d}$, respectively. Note that no specific range of $\widetilde{d}$ is fixed in the geometry histogram, while the range of the intensity values in [0, 255] in the histograms of typical images. Therefore, a user can adaptively select the dynamic range of $[\widetilde{d}_{min}, \widetilde{d}_{max}]$ according to the stretching level of geometry histogram. In practice, we determine $\widetilde{d}_{min}$ and $\widetilde{d}_{max}$ as

$$\begin{aligned} \widetilde{d}_{min} &= \tau\, d_{min} \\ \widetilde{d}_{max} &= \tau\, d_{max} \end{aligned} \tag{3.11}$$

where τ is the scaling parameter, and d_{min} and d_{max} mean the lower bound and the upper bound of d in the original geometry, respectively.

3.5 Normal Mapping Based Geometry Contrast Enhancement

Although our approach is trustworthy to improve the visibility of the weak features in a 3D point model, moving point positions yields an undesired artifact. As shown in Figure 3.13, we can observe discontinuities around the modified surfaces when rendering the enhanced model. This artifact becomes severer when closing up the modified surface regions.

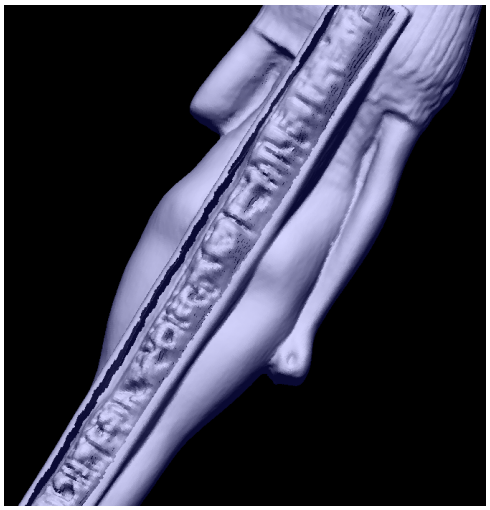


Figure 3.13: Discontinuity artifact in the enhanced model.

In order to alleviate this artifact, we adopt a concept of normal mapping (also known as bump mapping) (Blinn 1978, Cohen et al. 1998). The normal mapping is normally used to create the illusion of details without adding geometry to the 3D models. Such illusion effect of normal mapping can be performed by putting a normal map containing 2D height information on the simple and smooth surface, which finally represents the wrinkled surface as illustrated in Figure 3.14. For example, this faking technique is used to improve the appearance and details of a simple mesh model by producing a normal map from a highly detailed mesh model (*Normal mapping --- Wikipedia, The Free Encyclopedia* 2012). It is generally agreed that the normal mapping is same with the bump mapping, where the bump mapping uses single-axis normal map, the normal mapping exploits three-axis normal maps to be stored as color images having RGB channels.

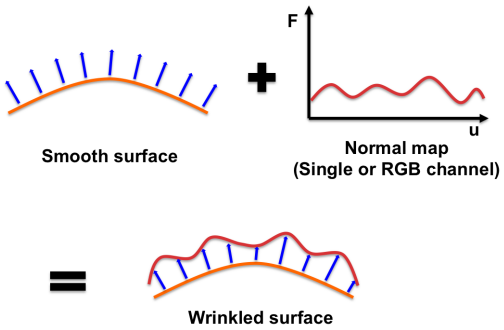


Figure 3.14: Normal mapping.

Motivated by the concept of normal mapping, we introduce a novel technique to alleviate the discontinuity artifact. We first define a mapping function $\psi(\mathbf{p})$ for each weak feature point, which is given as

$$\psi(\mathbf{p}) = \overrightarrow{\mathbf{n}_{\mathbf{p}}} \tag{3.12}$$

where $\overrightarrow{\mathbf{n}_{\tilde{\mathbf{p}}}}$ denotes the normal estimated from the new position $\tilde{\mathbf{p}}$ of $\mathbf{p}$. As shown in Figure 3.15, the estimated normal $\overrightarrow{\mathbf{n}_{\tilde{\mathbf{p}}}}$ at the new position $\tilde{\mathbf{p}}$ is assigned to the corresponding original point $\mathbf{p}$ by the mapping function $\psi(\mathbf{p})$. Note that this procedure is performed only on the weak feature points.

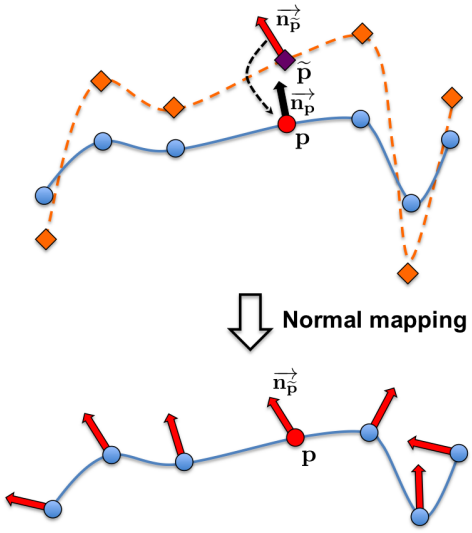


Figure 3.15: Normal mapping function.

3.6 Fast Processing

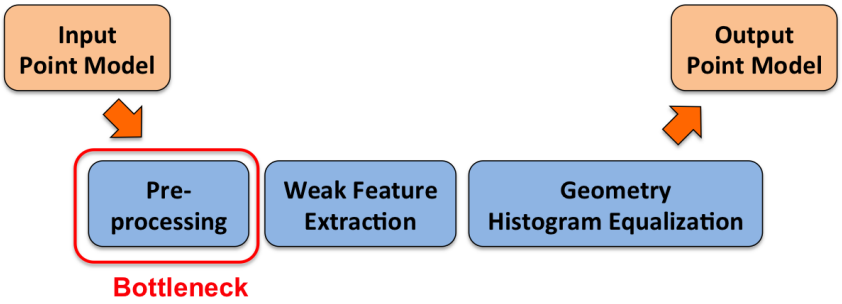


Figure 3.16: Bottleneck in the preprocessing step.

Order	Operation	Processing Time
1	Memory loading	400 msec
2	Preprocessing	3700 msec
3	Weak Feature Extraction	200 msec
4	Geometry histogram Equalization	700 msec

Table 3.2: Comparison of processing time for each operation in the proposed algorithm. (Input: the "Golf-club" model composed of 209,779 points.)

As shown in Figure 3.16 and Table 3.2, the preprocessing stage of the proposed algorithm yields the largest amount of processing time when compared to the other stages. Especially, finding KNN requires

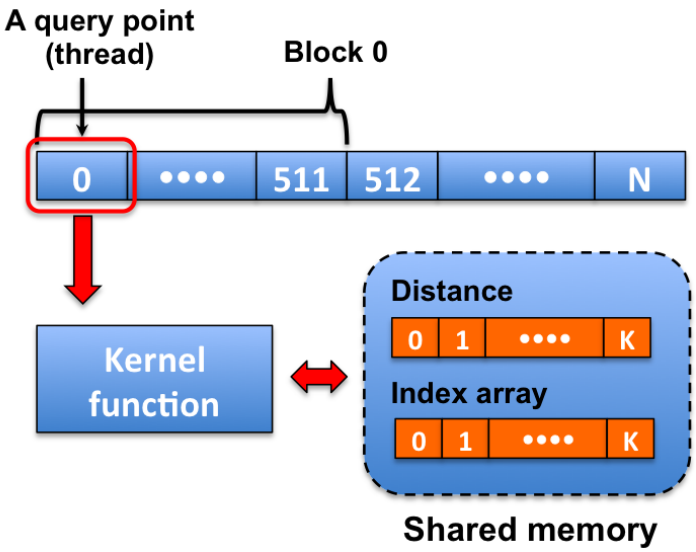


Figure 3.17: Flow timeline of KNN algorithm based on GPU architecture.

many basic operations and the KNN search area polynomially grows according to the number of points (Garcia et al. 2008, Hwang & Wen 1998). The generic CPU is not optimized for dealing with such time-consuming tasks. Some well-optimized CPU based KNN algorithms (Arya et al. 1998) have been presented in order to reduce the computation time. Even though these are fairly competitive, they cannot efficiently handle very huge data sets because the distance computation on the generic single-core CPU is performed in serial order.

We present an alternative method for reducing the computation time. Since the KNN computation can be parallelizable, we implement this algorithm based on the GPU architecture (Garcia et al. 2008). For the implementation of the KNN algorithm on the GPUs, we need to build KD-trees in the CPU based system, then copy the tree information from host memory to device memory on the GPUs. As shown in Figure 3.17, a number of basic operations of KNN searching can be performed in parallel in GPU architecture. Hence, each computation to find the nearest neighbors at each point can be implemented per thread. Then each thread calls the kernel function composed of the KNN algorithm. In the middle of the KNN computation per thread, we especially allocate shared memory on the GPUs in order to store the indices and corresponding positions of neighboring points. Since the shared memory is very fast device memory, it allows to reduce memory access time.

Chapter IV

Experimental Results

4.1 Experimental Environment

The proposed algorithm was performed in the Windows 7 platform with Intel Core i7 2.79 GHz CPU, 6GB RAM, Nvidia Quadro FX 1800 GPU. We have also used the glDrawArrays API of OpenGL to facilitate a simple and fast rendering of points (Wright 2010). For the implementation of the GPU based KNN algorithm, we have applied the parallel programming procedure to the KD-tree construction by using CUDA (Compute Unified Device Architecture) language (Nvidia 2011). In addition, we have also attempted to examine another well-optimized KNN libraries such as FLANN (Fast Library for Approximate Nearest Neighbors) (Muja 2011) and Google's PCL (Point Cloud Library) (Rusu & Cousins 2011), in order to evaluate our application on the GPU architecture. The following table denotes the specification of the test 3D point models used to evaluate the performance of the proposed method.

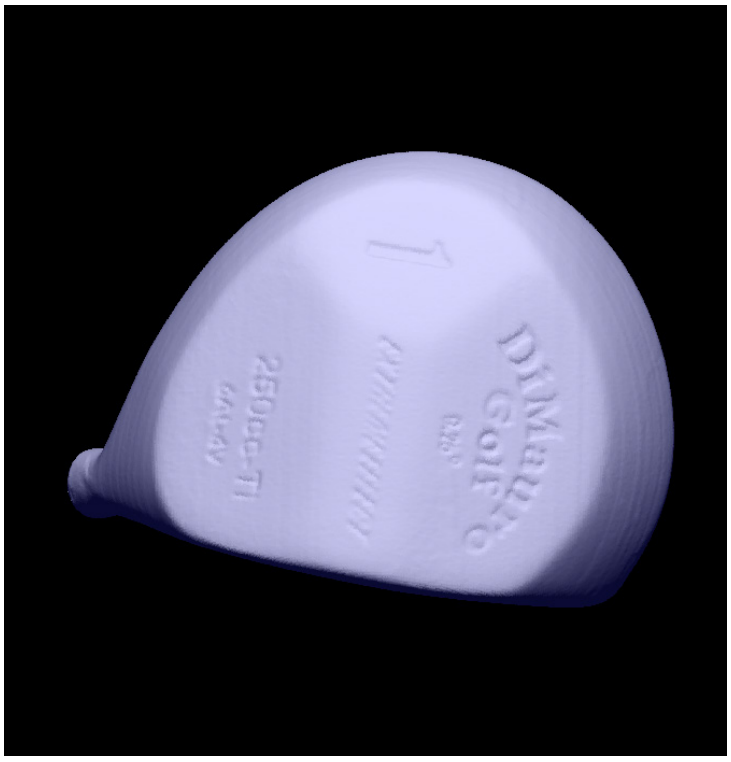
Input model	Number of point	Frames per second
Golf-club	209,779	43 fps
Isis	187,644	46 fps
Screwdriver	27,152	50 fps
RockerArm	40,177	49 fps

Table 4.1: Specification of test 3D point models.

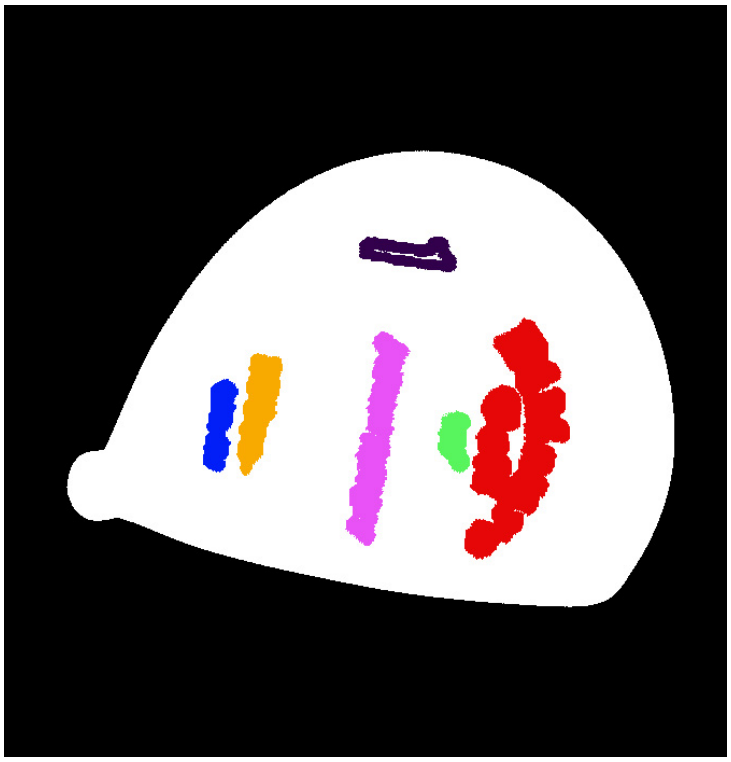
4.2 Experimental Results

Figure 4.1 shows the performance of feature extraction on the "Golf-club" model. The detected weak feature points are locally grouped into the several regions with different colors. Figure 4.2a and Figure 4.2b show the geometry contrast enhanced models without and with the region growing method to determine the weak feature regions. While the rough and coarse edges are shown around the weak features in Figure 4.2a, the weak feature regions are more clearly recognized in Figure 4.2b.

Figure 4.3, 4.4, 4.5, and 4.6 show the performances of the geometry contrast enhancement algorithm on the "Golf-club", "Isis", "Screwdriver", and "RockerArm" models, respectively. Figure 4.3b directly renders the modified enhanced geometry model and Figure 4.3c shows the normal mapping based enhancement result. We observe that the shallowly carved letters and numbers on the surface of the original model are clearly represented in the enhancement results. Figure 4.4a indicates the "Isis" model, where the indistinct patterns are engraved on the back of the model. As shown in Figure 4.4b, we can more easily visualize the patterns by enhancing the geometry contrast. Moreover, we observe that the discontinuity artifact yielded by direct rendering of the enhanced model is alleviated in Figure 4.4c due to the normal mapping. In Figure 4.5b and 4.5c, we can exaggerate the patterns which are shallowly engraved on the bottom of the "Screwdriver" model. Similarly, Figure 4.6b and 4.6c show that the dim patterns on the surface of the original "RockerArm" model are clearly observed in the enhanced model.

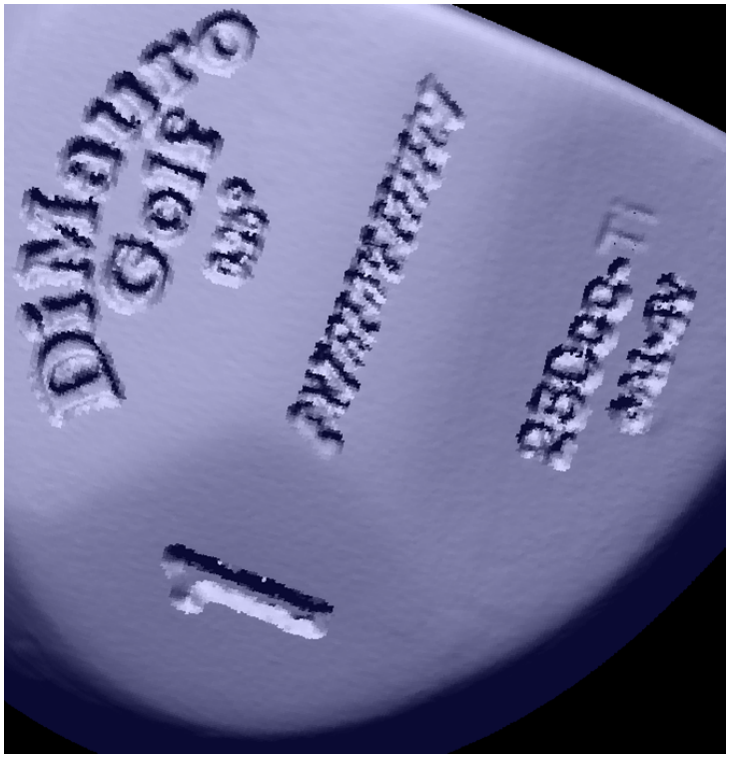


(a)



(b)

Figure 4.1: Weak feature extraction. (a) The "Golf-club" model. (b) Color-labeled weak feature regions.



(a)

(b)

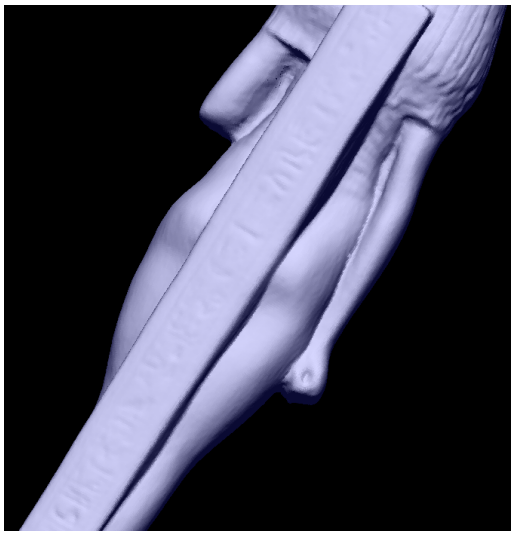
Figure 4.2: Geometry contrast enhancement on the "Golf-club" model (a) without the region growing method and (b) with the region growing method, respectively.

(a)

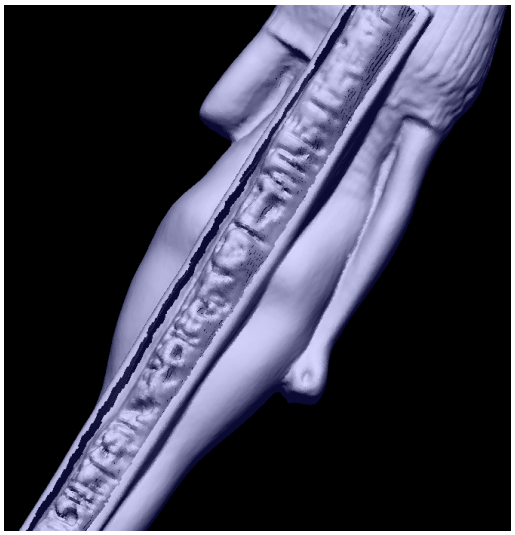
(b)

(c)

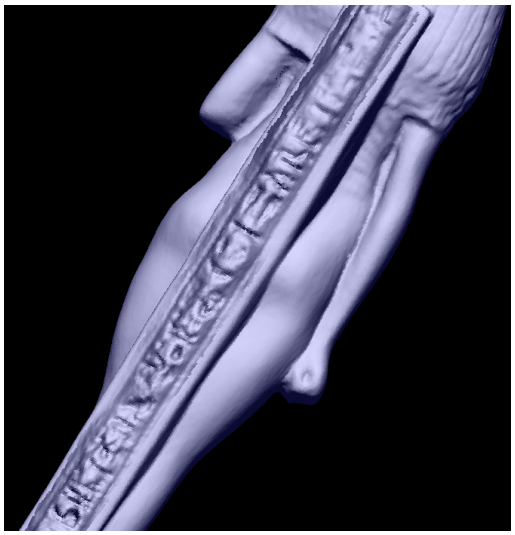
Figure 4.3: Results of the geometry contrast enhancement where $\tau = 7$. (a) Original "Golf-club" model. Enhanced models (b) without the normal mapping and (c) with the normal mapping.



(a)

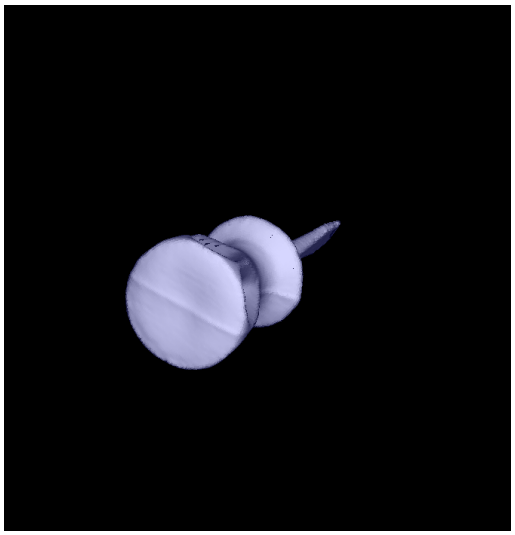


(b)



(c)

Figure 4.4: Results of the geometry contrast enhancement where $\tau = 7$. (a) Original "Isis" model. Enhanced models (b) without the normal mapping and (c) with the normal mapping.

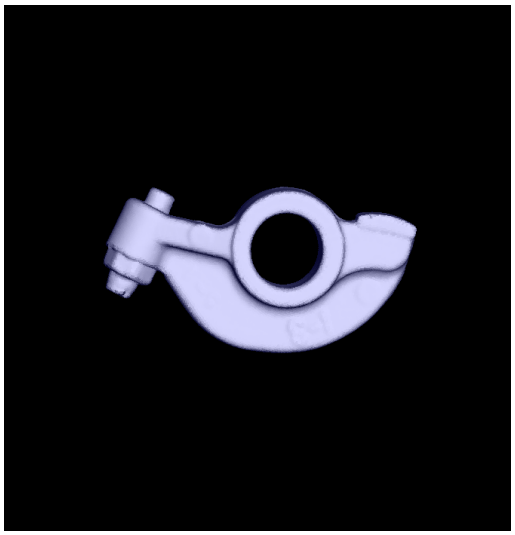


(a)

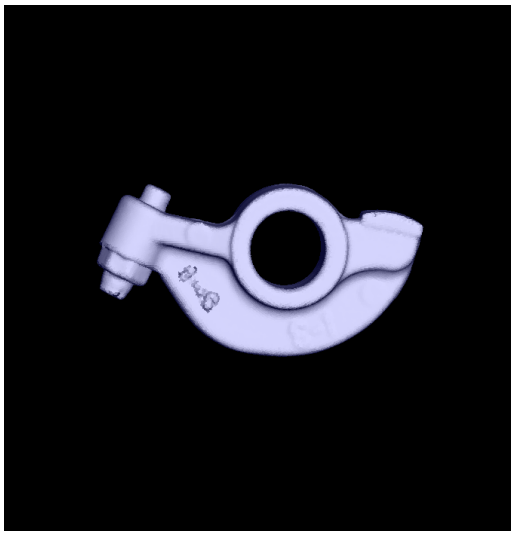
(b)

(c)

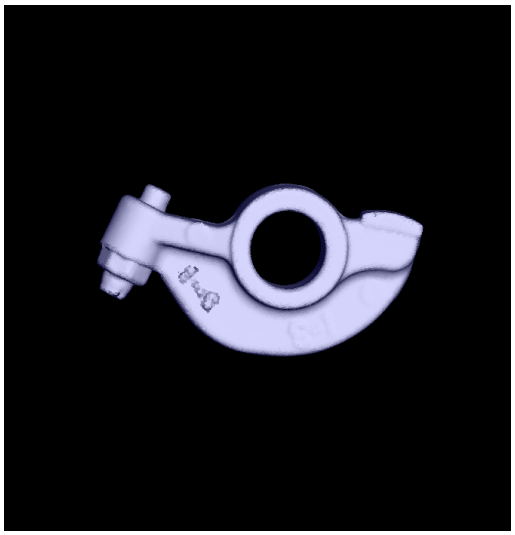
Figure 4.5: Results of the geometry contrast enhancement where $\tau = 7$. (a) Original "Screwdriver" model. Enhanced models (b) without the normal mapping and (c) with the normal mapping.



(a)



(b)



(c)

Figure 4.6: Results of the geometry contrast enhancement where $\tau = 14$. (a) Original "RockerArm" model. Enhanced models (b) without the normal mapping and (c) with the normal mapping.

Chapter V

Conclusion

In this thesis, we introduced a novel concept of geometry contrast and proposed a geometry contrast enhancement algorithm for 3D point models using the histogram modification techniques. In weak feature extraction, we defined the points as the weak feature points when the normal deviation belongs to a specific range. Then, a region growing method is applied to generate the locally grouped weak feature regions.

We defined the geometry histogram for each weak feature region as the distribution of the points along the normal direction to a locally approximated plane. By equalizing and stretching the distribution of points, we increased the visibility of 3D point models and enhanced the geometry contrast. Moreover, we presented an alternative rendering method using normal mapping without directly changing the original point positions.

Although the proposed method was trustworthy for 3D point models that have low geometry contrast, two major challenges are still remained. One is that the proposed method is not adaptive because we need to set some user defined parameters whenever the input model changes. The other is that we observe that the unnatural phenomenon cannot be eliminated even if we apply normal mapping based rendering, since the concept of normal mapping is not enough for implementing the self-occlusion effect.

References

Arya, S., Mount, D., Netanyahu, N., Silverman, R. & Wu, A. (1998), `An optimal algorithm for approximate nearest neighbor searching fixed dimensions', *Journal of the ACM (JACM)* **45**(6), 891--923.

Bentley, J. (1975), `Multidimensional binary search trees used for associative searching', *Communications of the ACM* **18**(9), 509--517.

Blinn, J. (1978), `Simulation of wrinkled surfaces', **12**(3), 286--292.

Cohen, J., Olano, M. & Manocha, D. (1998), Appearance-preserving simplification, *in* `Proceedings of the 25th annual conference on Computer graphics and interactive techniques', ACM, pp. 115--122.

Demarsin, K., Vanderstraeten, D., Volodine, T. & Roose, D. (2007), `Detection of closed sharp edges in point clouds using normal estimation and graph theory', *Computer-Aided Design* **39**(4), 276--283.

Fang, J., Fang, S., Huang, J. & Tuceryan, M. (2006), Digital geometry image analysis for medical diagnosis, *in* `Proceedings of the 2006 ACM symposium on Applied computing', ACM, pp. 217--221.

Garcia, V., Debreuve, E. & Barlaud, M. (2008), Fast k nearest neighbor search using gpu, *in* `Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on', Ieee, pp. 1--6.

Gonzalez, R. C. & Woods, R. E. (2007), *Digital image processing*, Prentice Hall.

Gumhold, S., Wang, X. & MacLeod, R. (2001), `Feature extraction from point clouds', pp. 293--305.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. & Stuetzle, W. (1994), Surface reconstruction from unorganized points, PhD thesis, University of Washington.

Hwang, W. & Wen, K. (1998), `Fast knn classification algorithm based on partial distance search', *Electronics letters* **34**(21), 2062--2063.

Jackins, C. & Tanimoto, S. (1980), `Oct-trees and their use in representing three-dimensional objects', *Computer Graphics and Image Processing* **14**(3), 249--270.

Levoy, M. (1999), The digital michelangelo project, *in* `3-D Digital Imaging and Modeling, 1999. Proceedings. Second International Conference on', IEEE, pp. 2--11.

Muja, M. (2011), `Flann, fast library for approximate nearest neighbors'. `http://mloss.org/software/view/143/`.

Nixon, M. & Aguado, A. (2008), *Feature extraction and image processing*, Academic Press.

Normal mapping --- Wikipedia, The Free Encyclopedia (2012), `http://en.wikipedia.org/wiki/Normal_mapping`.

Nvidia, C. (2011), `Nvidia cuda c programming guide', *NVIDIA Corporation* .

Pauly, M., Keiser, R. & Gross, M. (2003), `Multi-scale feature extraction on point-sampled surfaces', **22**(3), 281--289.

Rusu, R. & Cousins, S. (2011), 3d is here: Point cloud library (pcl), *in* `Robotics and Automation (ICRA), 2011 IEEE International Conference on', IEEE, pp. 1--4.

Weber, C., Hahmann, S. & Hagen, H. (2010), Sharp feature detection in point clouds, *in* `Shape Modeling International Conference (SMI), 2010', IEEE, pp. 175--186.

Wright, R. S. e. a. (2010), *OpenGL SuperBible*, Addison-Wesley.

Acknowledgements

Foremost, I would really like to express my deep and sincere gratitude to my advisor Prof. Jae-Young Sim for giving me the opportunity to be his first graduate student and the continuous support of my M.S. study and research. His wide knowledge and logical way of thinking have been of great value for me. Moreover, His guidance continuously helped me in research and writing of my thesis. I could not have imagined having a better advisor and mentor for my M.S. thesis.

Besides my advisor, I wish to express my warm and sincere thanks to the rest of my thesis committee members which include Prof. Seoung-Joon Yang and Won-Ki Jeong, for their encouragement, insightful comments and evaluation. Their ideas and concepts have had a remarkable influence on my entire career in the field of visual information processing research.

I would like to thank my fellow labmates in Visual Information Processing Laboratory: Tae-Hui Yun and Su-Yeong Kim, for giving me such a pleasant time while working together for last one and a half year. Also I would like to thank my dear friends in UNIST: Heung-Seok Ko and Hee-Cheol Hwang. I am thankful to Kang-Yeoul Lee, Min-Wook Kwak, Kyu-Yeoul Shin, Young-Hoon Jeong and Seong-Suk Seo, for a comfortable atmosphere in my unist journey.

Last but not the least, I would like to thank my family: my parents Jong-Cheol Nam and Bong-Sun Yang for giving birth to me and supporting me spiritually throughout my life. They were always cheering me up and stood by me through the good and bad times. In addition, I would especially like to thank Chang-Woo Nam who is my permanent mentor as well as my only elder brother. I would never have been able to accomplish many challenging tasks without his assistances and inspiration.

Finally, I would like to thank everybody who contributed to the successful completion of this thesis.

Thank you Jesus, blessed be your name!